(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 March 2002 (14.03.2002)

PCT

(10) International Publication Number
WO 02/21264 A2

(51) International Patent Classification[7]: G06F 9/00

(21) International Application Number: PCT/US01/28146

(22) International Filing Date:
6 September 2001 (06.09.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/655,751 6 September 2000 (06.09.2000) US

(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, M/S: UPAL01-521, Palo Alto, CA 94303 (US).

(72) Inventors: CONNORS, James, T.; 2707 Sylvia Drive, North Bellmore, NY 11710 (US). ELLIS, Craig, S.; 2 Executive Drive, Hauppauge, NY 11788 (US).

(74) Agent: OLYNICK, Mary, R.; BEYER WEAVER & THOMAS, LLP, P.O. Box 778, Berkeley, CA 94704-0778 (US).

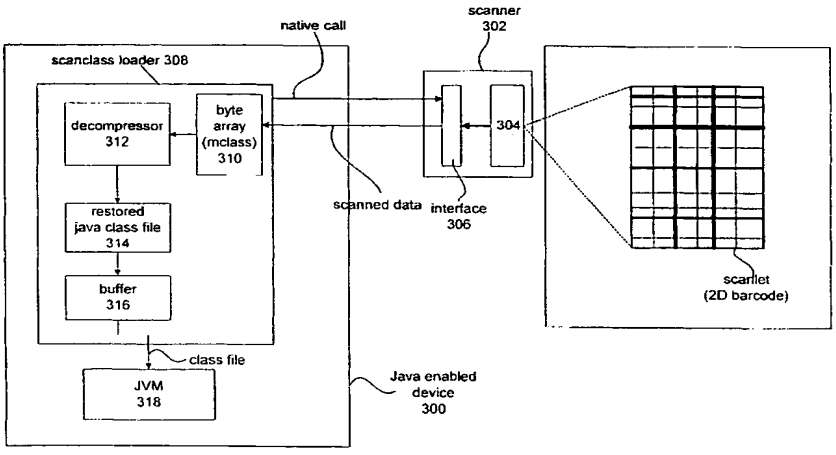
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: METHOD AND APPARATUS FOR REPRESENTING EXECUTABLE CONTENT WITHIN A BARCODE (SCANLET)



(57) Abstract: Systems and methods for storing executable content in an icon, such as a scanlet, and using the stored executable content are described. In general, in order to store executable content, such as a Java class file, a determination is made whether or not the Java class file to be stored can be adequately compressed so as to fit within the scanlet. Once so determined, the Java class file is losslessly compressed and encoded to form the scanlet. In order to execute the executable content incorporated in the scanlet, in one embodiment, a conventional scanner reads the scanlet and sends the data in the form of mclass data bytes to a buffer or other such appropriate storage device. A decompressor coupled to the buffer then decompresses the mclass data bytes stored in the buffer to form a restored Java class file that replicates the original Java class file. The restored Java class file is then used to provide the executable content for a Java Virtual Machine incorporated in the Java enabled device.

WO 02/21264 A2

PATENT APPLICATION

METHOD AND APPARATUS FOR REPRESENTING EXECUTABLE CONTENT WITHIN A BARCODE (SCANLET)

BACKGROUND OF THE INVENTION

1. Field of Invention

The invention relates generally to computing and information systems. More particularly, methods and apparatus for representing executable content in the form of a barcode, also referred to as a scanlet.

2. Description of Relevant Art

One of the many advantages of Java is that it can also be used to build small application modules, or applets, that can be embedded in any number and type of documents, such as a Web page. A Java applet is a small program that can be sent along with a Web page to a user that can perform interactive animations, immediate calculations, or other simple tasks without having to send a user request back to the server. As an example, as shown in Fig. 1, a distributed computer system 100 includes a client computer 102 that is coupled to a server (host) computer 104. The computer 102 includes a browser application 106 that, in turn, includes a requested Web page 108 having an applet 110 embedded therein capable of performing various tasks. In most situations, the applet 110 is executed by a Java Virtual Machine (JVM) 112 that in this example is also resident in the browser 106.

In order for the JVM 112 to execute the applet 110, the applet's requisite component files (such as ".class files", images and sounds) represented by files 114 -

118 must be downloaded from the server 104 to the JVM 112. Typically the server 104 is part of a distributed network of computers, such as the Internet, or in some cases could be part of an intranet type of arrangement. In any case, the files 114-118 that are required for the JVM 112 to run the applet 110 include Java class files as well as resource files that are used to support the execution of the applet 110. Such class files, includes a main class file, main.class 114, that is used by the JVM 112 as an entry point for execution of the applet 110. The server 104 also stores other class files such as b.class 116 that are used by the JVM 112 in the furtherance of executing the applet 110. Various image and sound components used in the execution of the applet 110 are stored in resource files such as c.image 118.

Other approaches to storing executable content (such as Java class files) besides those involving server side storage techniques have also been attempted. Once such approach is the use of a barcode, the content of which has traditionally been relatively static in nature. Unfortunately, however, although promising, the static nature (and small storage capacity) of one dimensional (1D) bar code precludes it as an effective way of storing executable content. However, with the advent of two dimensional (2D) barcodes and two dimensional scanning technology with its resultant larger data storage capability, the possibility of storing executable content (such as, for example, a Java class file) within a two dimensional bar code has become feasible.

Currently, there are a myriad of barcode symbologies available, each having differing capabilities. The current trade offs between these various formats appear to revolve around data density versus reliability and error correction. One such barcode symbology is represented by the PDF417 barcode standard selected by the Symbol

Technologies Inc. of Holtsville, NY (a recognized leader in scanning technology) which is a barcode standard understood by a large number of available 2D scanners.

Even though the PDF 417 accomodates much more information than does the ubiquitous 1D barcode, it is still inadequate to support anything but the most rudimentary of Java class files. Specifically, the PDF417 has a maximum capacity of 1080 bytes of binary data, whereas the class file for a simple program (such as the Java program "HelloWorld") requires up to 500 bytes, it is clear that as currently configured, a conventional 2D barcode could only provide a class file of sufficient size to support only the most simple programs.

Therefore, what is desired is the capability of incorporating executable content in a 2D barcode and using the executable content incorporated therein as input to a simple program.

SUMMARY OF THE INVENTION

Broadly speaking, the invention relates to an improved method, apparatus and computer system for encoding executable content in a 2D barcode to form a scanlet that is capable of storing sufficient executable content for a simple program. The invention can be implemented in numerous ways, including as a method, a computer system, and an apparatus. Several embodiments of the invention are discussed below.

According to one aspect of the present invention, a method for A method for symbolically encoding executable content in an icon is described. The executable content is converted to an uncompressed data file that is then compressed to form a compressed data file having data consistent with the executable content. The data included in the compressed data file is then encoded into the icon.

In another embodiment, a method of executing compressed executable content stored in the icon is disclosed. The compressed executable content is read and uncompressed. The uncompressed executable content is then input to a program which performs the executable content.

These and other advantages of the present invention will become apparent upon reading the following detailed descriptions and studying the various figures of the drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

The invention, together with further advantages thereof, may best be understood by reference to the following description taken in conjunction with the accompanying drawings in which:

Fig. 1 shows a conventional distributed computer system that includes a client computer coupled to a server (host) computer.

Fig. 2 is a system for compressing a Java class file to form a scanlet in accordance with an embodiment of the invention.

Fig. 3 shows a Java enabled device for reading a scanlet and executing the Java class file encoded therein in accordance with an embodiment of the invention.

Fig. 4 shows a flowchart detailing a process for encoding a scanlet in accordance with an embodiment of the invention.

Fig. 5 is a flowchart detailing a process for reading and executing data stored in a scanlet by a Java enabled device in accordance with an embodiment of the invention.

Fig. 6 illustrates a computer system that can be employed to implement the present invention.

DETAILED DESCRIPTION OF THE EMBODIMENTS

The following description is provided to enable any person skilled in the art to make and use the invention and sets forth the best modes contemplated by the inventor for carrying out the invention. Various modifications, however, will remain readily apparent to those skilled in the art, since the basic principles of the present invention have been defined herein specifically to provide a novel protocol and apparatus for storing executable content in the form of a 2D barcode.

Reference will now be made in detail to a preferred embodiment of the invention. An example of the preferred embodiment is illustrated in the accompanying drawings. While the invention will be described in conjunction with a preferred embodiment, it will be understood that it is not intended to limit the invention to one preferred embodiment. To the contrary, it is intended to cover alternatives, modifications, and equivalents as may be included within the spirit and scope of the invention as defined by the appended claims.

In the following description, frameworks and methods of storing executable content and using the stored executable content are described. The invention will initially be described in terms of a Java enabled device but can be used in any device capable of performing executable content, such as an instruction set. In general, in order to store executable content, such as a Java class file, in the form of a 2D barcode, a determination is made whether or not the Java class file to be stored can be adequately compressed so as to fit within the available barcode technology. Once so determined, the Java class file is compressed in such a manner as to be fully comprehended by the 2D barcode in the form of a scanlet. It should be noted that the compression methodology is substantially a lossless compression methodology whereby there is no loss in executable content from the Java class file. Once the

original Java class file has been compressed to form what is referred to as an mclass file, the mclass file is copied, decompressed and checked against the original Java class file in order to assure fidelity between the original Java class file and the mclass file.

Using conventional printing techniques, the mclass file is then used to form the scanlet suitably arranged to provide executable content (in the form of the decompressed mclass file) to any Java enabled device.

In order to execute the executable content incorporated in the scanlet, in one embodiment, a conventional scanner reads the scanlet and sends the data in the form of mclass data bytes to a buffer or other such appropriate storage device. A decompressor coupled to the buffer then decompresses the mclass data bytes stored in the buffer to form a restored Java class file that replicates the original Java class file. The restored Java class file is then used to provide the executable content for a Java Virtual Machine incorporated in the Java enabled device.

Typically, computer programs written in the Java programming language are compiled into bytecodes or Java virtual machine instructions that are then executed by a Java virtual machine. The bytecodes are stored in class files that are input into the Java virtual machine for interpretation. Java source code includes the classic "Hello World" program written in Java. The source code is then input into a bytecode compiler that compiles the source code into bytecodes. The bytecodes are virtual machine instructions as they will be executed by a software emulated computer. Typically, virtual machine instructions are generic (i.e., not designed for any specific microprocessor or computer architecture) but this is not required. The bytecode compiler outputs a Java class file that includes the bytecodes for the Java program.

The Java class file is input into a Java virtual machine. The Java virtual machine is an interpreter that decodes and executes the bytecodes in the Java class file. The Java virtual machine is an interpreter, but is commonly referred to as a virtual machine as it emulates a microprocessor or computer architecture in software (e.g., the microprocessor or computer architecture that may not exist in hardware).

More recently, the Java programming language, an object-oriented language, has introduced the possibility of compiling output (called bytecode) that can run on any computer system platform for which a Java virtual machine (or bytecode interpreter) is provided. The Java virtual machine is designed to convert the bytecode into instructions that can be executed by the actual hardware processor. Using this virtual machine, rather than being interpreted one instruction at a time, bytecode can be recompiled at each particular system platform by, in some cases, a just-in-time (JIT) compiler. The operation of virtual machines or, more particularly, Java™ virtual machines, is described in more detail in The Java™ Virtual Machine Specification by Tim Lindholm and Frank Yellin (ISBN 0-201-63452-X), which is incorporated herein by reference.

The invention will now be described in terms of a Java enabled device such as for example, a personal digital assistant (PDA) having access to a barcode scanner, for example. It should be noted, however, that the invention is not limited to the described Java enabled device, nor is it limited to only the Java programming language. As a matter of fact, the invention can be used in any system where data is symbolically encoded which includes, but is not limited to, barcodes and the like.

Fig. 2 is a system 200 for compressing a Java class file to form a scanlet in accordance with an embodiment of the invention. The system 200 includes an mclass file compiler 202 arranged to convert a Java class file 204 into what is referred to as a

mclass file 205 suitable for being encoded in a barcode to form a scanlet 206. It should be noted that in order to maximize the number of Java class files which can be compressed sufficient for encoding as a scanlet, the smallest possible Java class file must be used. This can be accomplished in any number of ways. For instance, the "g:none" compiler option found in the Java 2 compilers produced by Sun Microsystems of Mountain View, CA can produce a class file that contains no debug information. In addition, it is possible to further reduce the size of the Java class file 204 by using other compiler optimization techniques and flags not mentioned here for sake of clarity but otherwise well known to those skilled in the art.

Once the Java class file 204 has been suitably optimized, the Java class file 204 is sent to an mclass pre-processor unit 208 arranged to further reduce the size of the Java class file 204. Once such pre-processor unit suitable for pre-processing the Java class file 204 is referred to as a Java obfuscator. A Java obfuscator, in addition to providing some security by making reverse engineering of the Java class file 204 more difficult, typically reduces the size of the Java class file 204.

In the described embodiment, the compiler 202 includes a pre-verifier unit 210 arranged to receive the pre-processed Java class file 204 and determine whether or not it can be compressed to a size suitable for forming the scanlet 206. Once the pre-verifier 210 has determined that the Java class file 204 can be sufficiently compressed to be encoded into the scanlet 206, it is sent to an mclass compressor unit 212 which forms the mclass file 205 by performing various compression algorithms on the pre-processed Java class file. Such compression algorithms include: examining constant pool entries and shrinking their component fields (if appropriate) from, for example, 16 bits to 8 bits; eliminating repetitive constant pool tags; introducing smaller mclass-specific bytecode instructions which can accommodate smaller offsets (such as 8 bit

signed offsets rather than 16 bit offsets); identifying and compressing null methods; and identifying commonly found strings and substituting them with appropriate abbreviations.

Once completed, the mclass file 205 is copied to a verifier unit 214 where it is compared to the original Java class file 204. This comparison assures that the compression performed by the mclass file compiler 202 was in fact a substantially lossless compression since it is important that no material information be lost in the compression process. Once verified against the original Java class file 204, the mclass file 205 is used by a conventional barcode printer 216 to form the scanlet 206.

Fig. 3 shows a Java enabled device 300 for reading a scanlet and executing the Java class file encoded therein in accordance with an embodiment of the invention. The device 300 includes, or is coupled to, a conventional scanner 302 having a lens 304 coupled to an interface 306 that receives instructions from a scanclassloader unit 308 included in the device 300. In the described embodiment, the scanclassloader unit 308 sends a native call to the interface 306 which results in the interface 306 sending the scanned data to a byte array 310 included in, or coupled to, the scanclassloader unit 308. The byte array 310 stores the mclass data bytes received from the scanner until such time as a decompressor 312 directs that the stored data bytes be transferred for decompressing. The decompressor 312 takes the mclass data bytes from the byte array 310 and expands them to form a restored Java class file 314 which is sent to class file buffer 316 until such time as it is required by a JVM 318 for execution.

Fig. 4 shows a flowchart detailing a process 400 for encoding a scanlet in accordance with an embodiment of the invention. The process 400 begins at 402 by pre-processing the Java class file by the pre-processor and at 404 by representing a

Java class file as a data structure. At 406, a determination is made by the pre-verifier whether or not the pre-processed Java class file can be compressed so as to be able to be encoded as a scanlet. If the pre-processed Java class file can not be adequately compressed, then the process 400 ends, otherwise a lossless compression is performed by the compressor unit at 408 to form an mclass file. A copy of the mclass file is forwarded to the decompressor at 410 where it is decompressed at 412 to form a restored Java class file. At 414 the restored Java class file is compared to the original Java class file and if it is determined at 415 that the comparison is not valid, then an error is thrown at 416, otherwise, control is passed to 418 where the verified mclass is sent to the barcode printer. At 420, the barcode printer generates the scanlet based upon the verified mclass file.

Fig. 5 is a flowchart detailing a process 500 for reading and executing data stored in a scanlet by a Java enabled device in accordance with an embodiment of the invention. The process 500 begins at 502 by a scanclassloader unit prompting a user to scan a scanlet that in the described embodiment takes the form of a 2D barcode. Once the barcode is scanned at 504, the mclass file from the scanned scanlet is input to a decompressor unit at 506. The decompressor unit then expands the mclass file at 508 and then verifies that the expanded mclass file is in fact a valid Java class file at 510. If the expanded mclass file is not a valid Java class file, then an error is thrown at 512, otherwise, control is passed to 514 where a JVM verifies and executes the bytecodes in the Java class file at 516.

Fig. 6 illustrates a computer system 600 that can be employed to implement the present invention. The computer system 600 or, more specifically, CPUs 602, may be arranged to support a virtual machine, as will be appreciated by those skilled in the art. As is well known in the art, ROM acts to transfer data and instructions uni-

directionally to the CPUs 602, while RAM is used typically to transfer data and instructions in a bi-directional manner. CPUs 602 may generally include any number of processors. Both primary storage devices 604, 606 may include any suitable computer-readable media. A secondary storage medium 608, which is typically a mass memory device, is also coupled bi-directionally to CPUs 602 and provides additional data storage capacity. The mass memory device 608 is a computer-readable medium that may be used to store programs including computer code, data, and the like. Typically, mass memory device 608 is a storage medium such as a hard disk or a tape which generally slower than primary storage devices 604, 606. Mass memory storage device 608 may take the form of a magnetic or paper tape reader or some other well-known device. It will be appreciated that the information retained within the mass memory device 608, may, in appropriate cases, be incorporated in standard fashion as part of RAM 606 as virtual memory. A specific primary storage device 604 such as a CD-ROM may also pass data uni-directionally to the CPUs 602.

CPUs 602 are also coupled to one or more input/output devices 610 that may include, but are not limited to, devices such as video monitors, track balls, mice, keyboards, microphones, touch-sensitive displays, transducer card readers, magnetic or paper tape readers, tablets, styluses, voice or handwriting recognizers, or other well-known input devices such as, of course, other computers. Finally, CPUs 602 optionally may be coupled to a computer or telecommunications network, *e.g.*, an Internet network, or an intranet network, using a network connection as shown generally at 612. With such a network connection, it is contemplated that the CPUs 602 might receive information from the network, or might output information to the network in the course of performing the above-described method steps. Such information, which is often represented as a sequence of instructions to be executed

using CPUs 602, may be received from and outputted to the network, for example, in the form of a computer data signal embodied in a carrier wave. The above-described devices and materials will be familiar to those of skill in the computer hardware and software arts.

Although only a few embodiments of the present invention have been described, it should be understood that the present invention may be embodied in many other specific forms without departing from the spirit or the scope of the present invention. By way of example, the scanclassloader can be used in any computing system capable of supporting the Java programming language.

Although the methods and apparatus of the invention are particularly suitable for implementation with respect to a Java™ based environment, the methods may generally be applied in any suitable object-based environment. In particular, the methods are suitable for use in platform-independent object-based environments. It should be appreciated that the methods may also be implemented in some distributed object-oriented systems. It should be appreciated that the present invention may generally be implemented on any suitable computing system having a compiler or any network of interconnected computers. Therefore, the present examples are to be considered as illustrative and not restrictive, and the invention is not to be limited to the details given herein, but may be modified within the scope of the appended claims along with their full scope of equivalents.

What is claimed is:

Claims:

1. A method for symbolically encoding executable content in an icon, comprising:

converting the executable content to an uncompressed data file;

compressing the uncompressed data file to form a compressed data file; and

encoding the compressed data file into the icon.

2. A method as recited in claim 1, wherein the icon is a two dimensional barcode.

3. A method as recited in claim 1, wherein the executable content is a Java class file.

4. A method as recited in claim 3, wherein the compressing, comprises:

examining a constant pool entry; and

shrinking a component field associated with the examined constant pool entry, if appropriate.

5. A method as recited in claim 3, wherein the compressing comprises:

eliminating repetitive constant pool tags.

6. A method as recited in claim 3, wherein the compressing comprises: introducing smaller mclass-specific bytecode instructions which can accommodate smaller offsets.

7. A method as recited in claim 3, wherein the compressing comprises:

identifying a null method; and

eliminating the identified null method.

8. A method as recited in claim 3, wherein the compressing comprises: identifying commonly a found string; and

substituting the found string with an appropriate abbreviation.

9. A method as recited in claim 1, further comprising:

copying the compressed data file;

de-compressing the compressed data file to form a restored data file; and

comparing the restored data file to the uncompressed data file.

10. A method as recited in claim 5, further comprising:

based on the comparing, if the restored data file matches the uncompressed data file, then

forwarding the compressed data file corresponding to the restored data file to an encoder.

11. A method of executing compressed executable content stored in an icon, comprising:

reading the compressed executable content;

uncompressing the read compressed executable content; and

inputting the uncompressed executable content to a program, wherein the program performs the executable content.

12. A method as recited in claim 11, wherein the executable content is a Java class file.

13. A method as recited in claim 12, wherein the icon is a 2D barcode.

14. A method as recited in claim 13, wherein the program is an applet.

15. A method as recited in claim 14, wherein the reading is performed by a barcode scanner.

16. A method as recited in claim 15, wherein the barcode scanner is coupled to a Java enabled device.

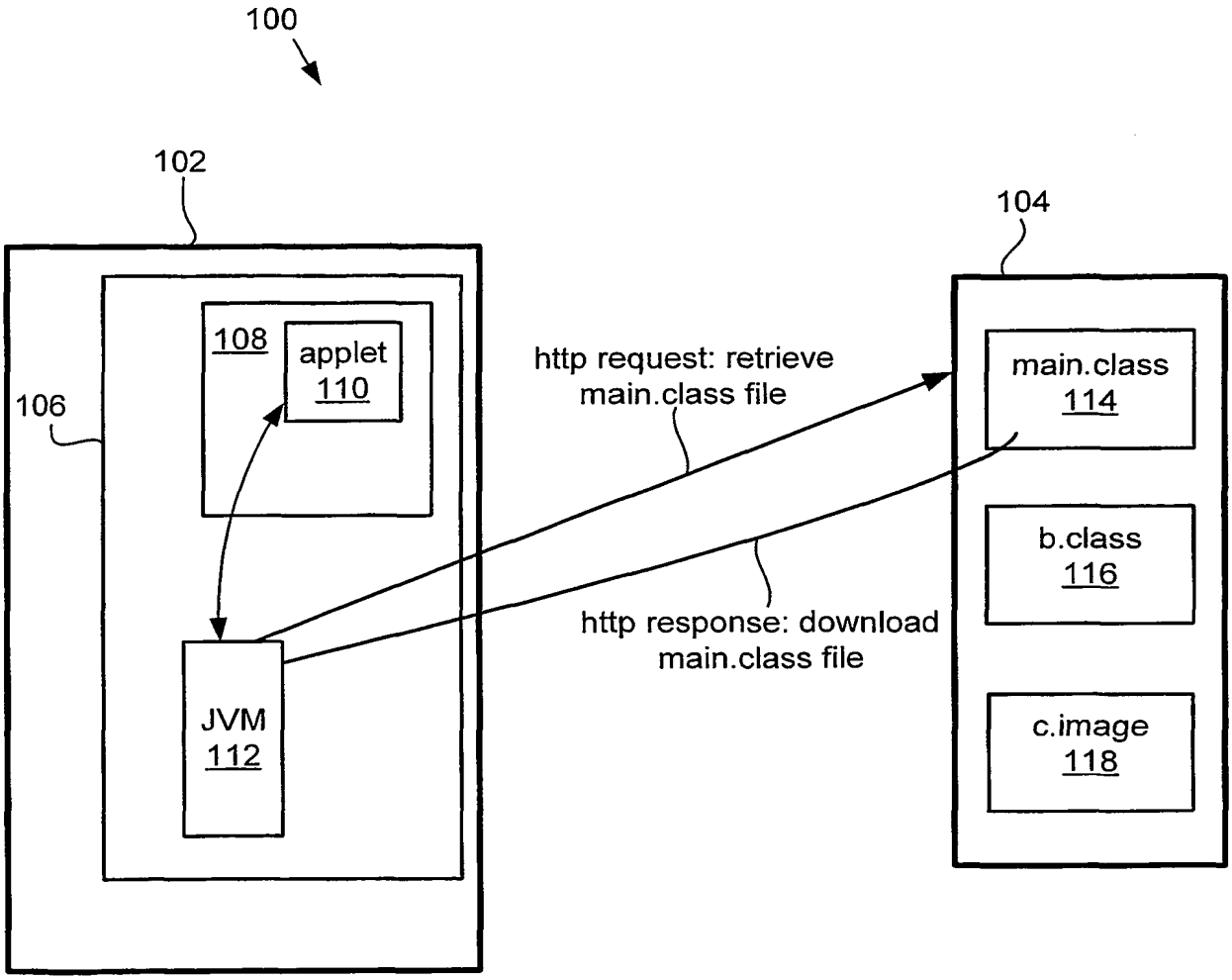


Fig. 1
Prior Art

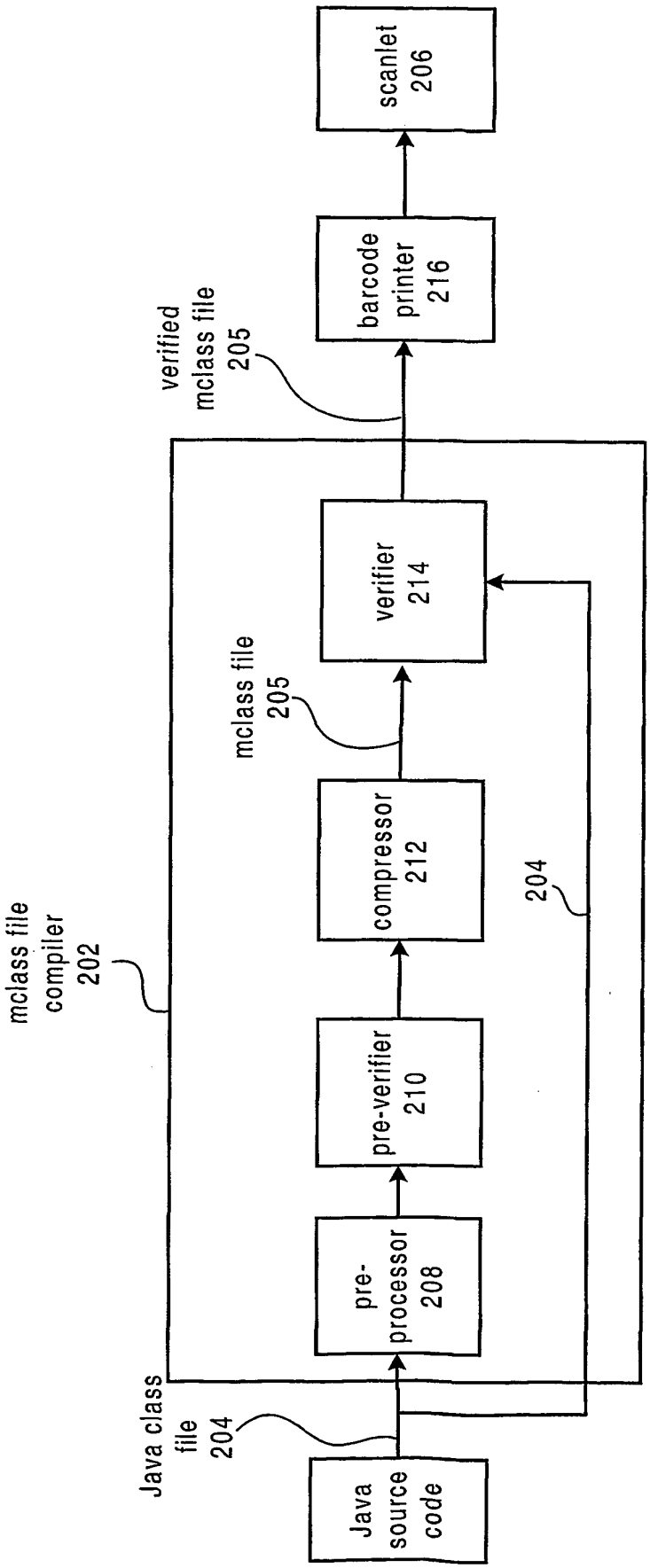


Fig. 2

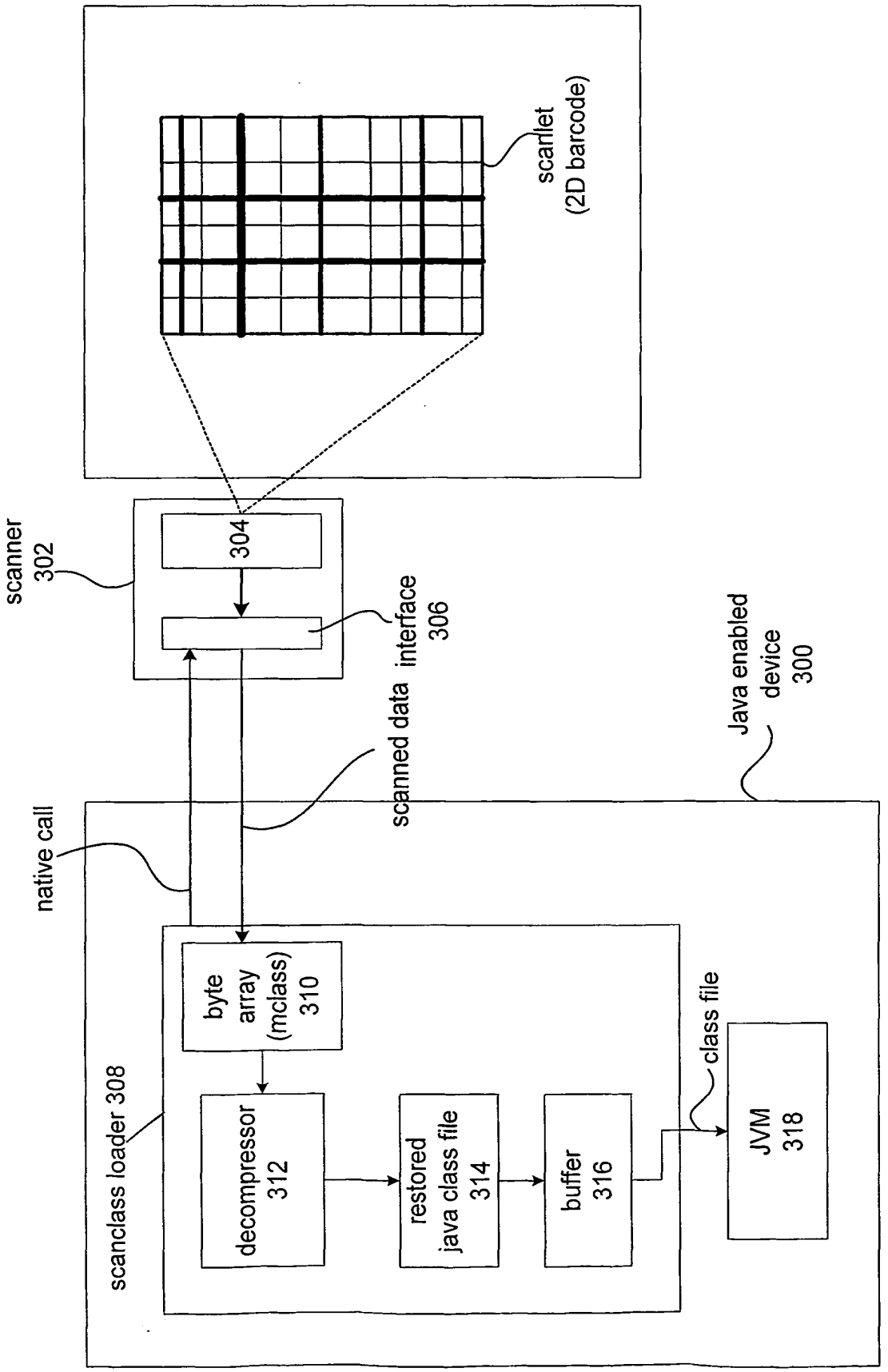


Fig. 3

Fig 4

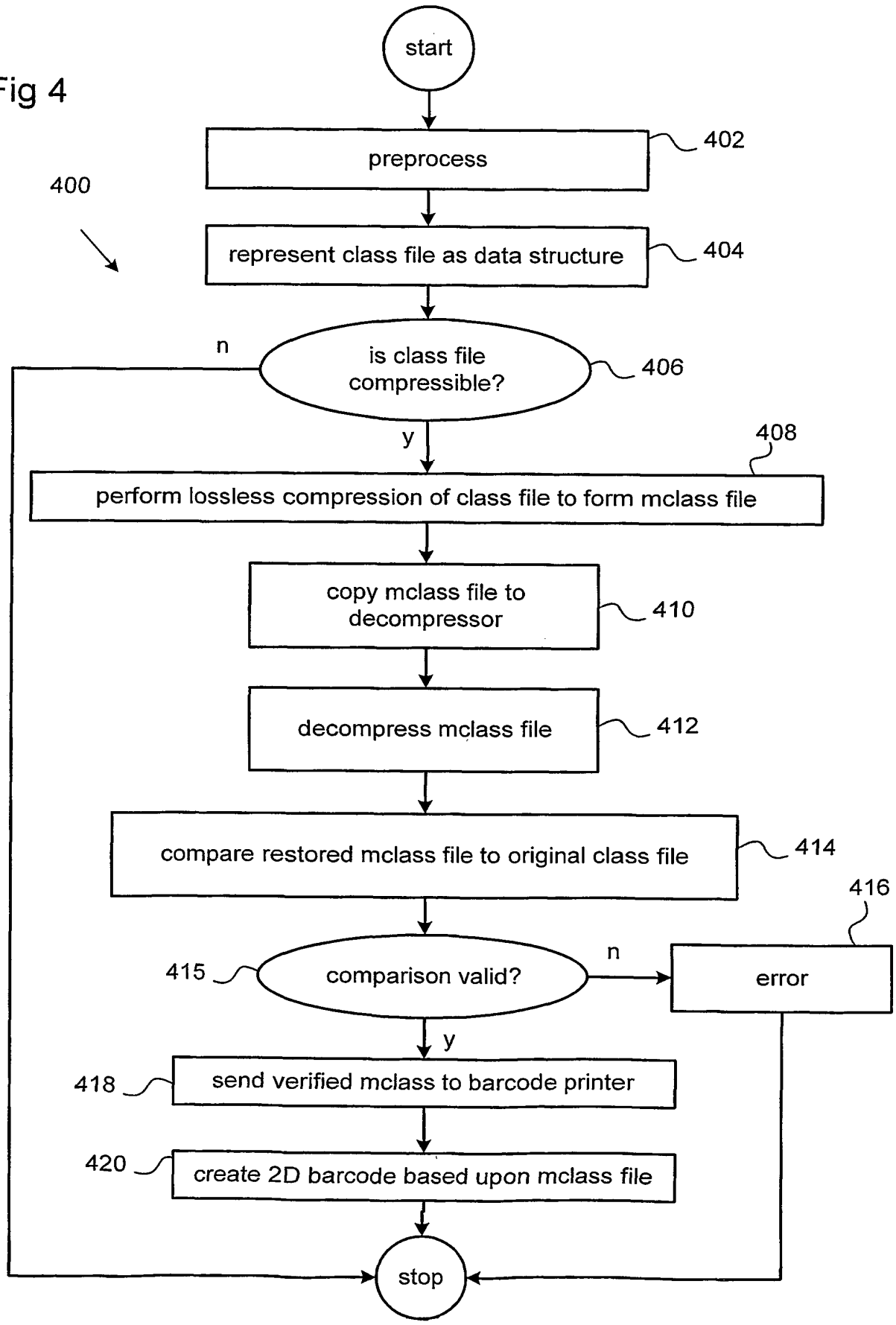
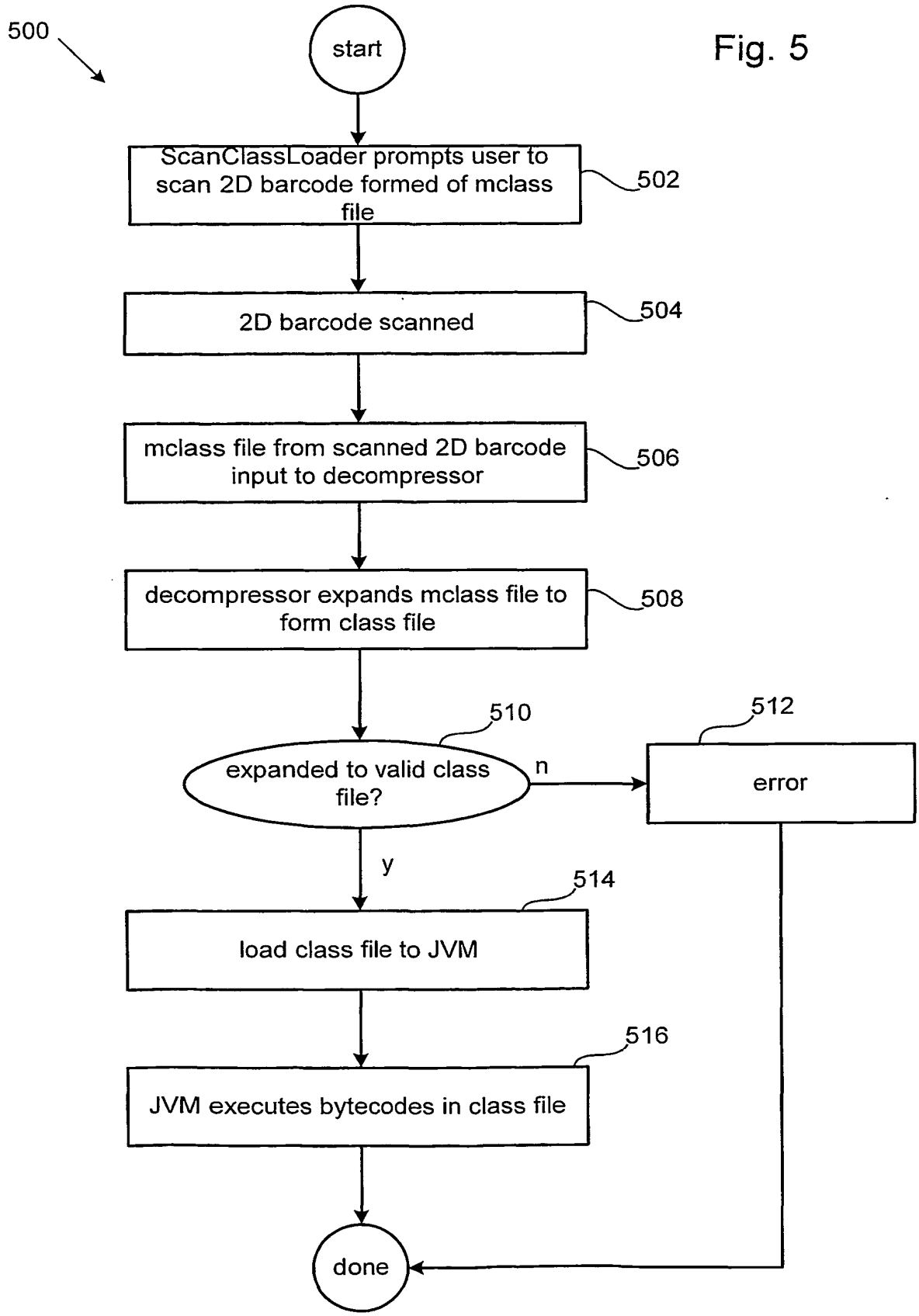


Fig. 5



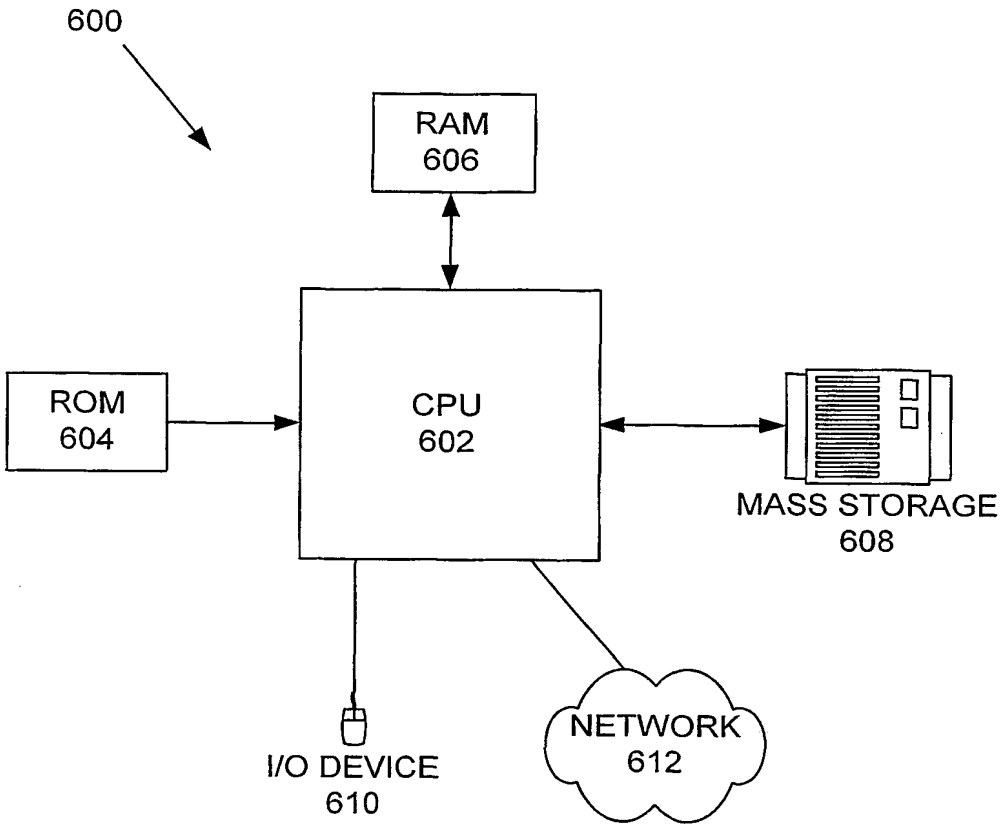


Fig. 6

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 March 2002 (14.03.2002)

PCT

(10) International Publication Number
WO 02/021264 A3

(51) International Patent Classification[7]: G06F 9/45

(21) International Application Number: PCT/US01/28146

(22) International Filing Date:
6 September 2001 (06.09.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/655,751 6 September 2000 (06.09.2000) US

(71) Applicant: **SUN MICROSYSTEMS, INC.** [US/US]; 901 San Antonio Road, M/S: UPAL01-521, Palo Alto, CA 94303 (US).

(72) Inventors: **CONNORS, James, T.**; 2707 Sylvia Drive, North Bellmore, NY 11710 (US). **ELLIS, Craig, S.**; 2 Executive Drive, Hauppauge, NY 11788 (US).

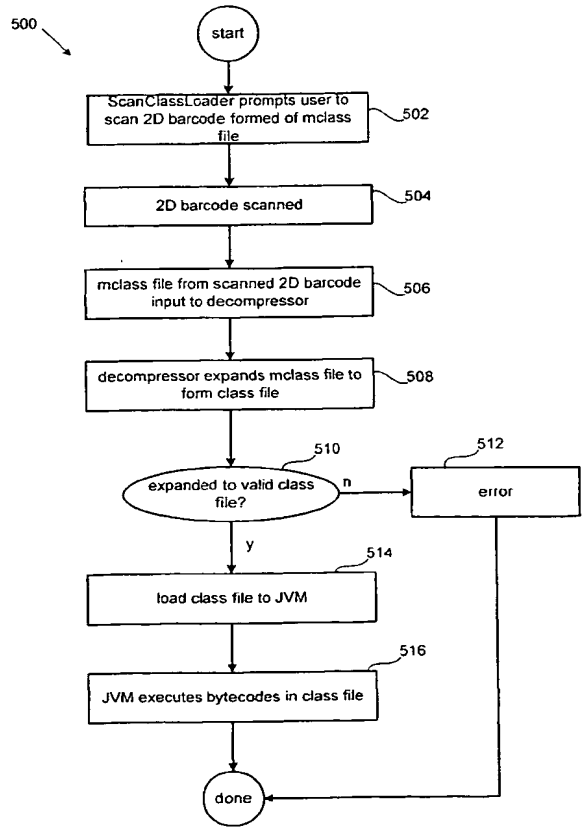
(74) Agent: **OLYNICK, Mary, R.**; BEYER WEAVER & THOMAS, LLP, P.O. Box 778, Berkeley, CA 94704-0778 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: METHOD AND APPARATUS FOR REPRESENTING EXECUTABLE CONTENT WITHIN A BARCODE (SCANLET)



(57) Abstract: Systems and methods for storing executable content in an icon, such as a scanlet, and using the stored executable content are described. In general, in order to store executable content, such as a Java class file, a determination is made whether or not the Java class file to be stored can be adequately compressed so as to fit within the scanlet. Once so determined, the Java class file is losslessly compressed and encoded to form the scanlet. In order to execute the executable content incorporated in the scanlet, in one embodiment, a conventional scanner reads the scanlet and sends the data in the form of mclass data bytes to a buffer or other such appropriate storage device. A decompressor coupled to the buffer then decompresses the mclass data bytes stored in the buffer to form a restored Java class file that replicates the original Java class file. The restored Java class file is then used to provide the executable content for a Java Virtual Machine incorporated in the Java enabled device.

WO 02/021264 A3

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

(88) Date of publication of the international search report:
1 August 2002

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 01/28146

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/45

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal, IBM-TDB, INSPEC, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 99 57885 A (KOCH ECKHARD ;MEDIASEC TECHNOLOGIES LLC (US); FRAUNHOFER CENTER FO) 11 November 1999 (1999-11-11)	1-3, 11-15
Y	abstract	4-10
X	US 5 837 986 A (BARILE JOHN ET AL) 17 November 1998 (1998-11-17)	1-3, 11-16
Y	abstract figure 1 column 2, line 44 -column 3, line 41 column 4, line 26 - line 40 column 8, line 6 - line 16 column 8, line 47 -column 9, line 27	4-8

-/--

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
16 May 2002	04/06/2002

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Écolivet, S.

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 01/28146

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
X	BERT MOORE: "A New Dimension in Bar Codes" INTERNET DOCUMENT, 'Online! December 1995 (1995-12), XP002199025 Retrieved from the Internet: <URL:http://www.byte.com/art/9512/sec7/art 3.htm> page 2, line 5 - line 9 ---	1,11
Y	CHARLES LEFURGY, TREVOR MUDGE: "Code Compression for DSP" PROCEEDINGS OF THE COMPILER AND ARCHITECTURE SUPPORT FOR EMBEDDED COMPUTING SYSTEMS (CASES 98) CONFERENCE, 'Online! 4 - 5 December 1998, XP002199141 George Washington University, Washington DC, États-Unis d'Amérique Retrieved from the Internet: <URL:http://www.eecs.umich.edu/{tnm/compre ss/publications/cse-tr-380-98.pdf> 'retrieved on 2002-05-16! page 1, left-hand column, line 1 -page 2, right-hand column, last line page 3, right-hand column, line 30 -page 4, left-hand column, last line ---	4-8
Y	ALFRED V. AHO, RAVI SETHI, JEFFREY D. ULLMAN: "Compilers -- Principles, Techniques, and Tools" 1987 , ADDISON-WESLEY SERIES IN COMPUTER SCIENCE , ÉTATS-UNIS D'AMÉRIQUE XP002199144 ISBN: 0-201-10088-6	4-8
A	page 554, line 4 -page 557, line 30 page 585 -page 609 ---	1-3
Y	MADLER@ALUMNI.CALTECH.EDU: "tired of gilbert" INTERNET DOCUMENT, 'Online! 6 August 1996 (1996-08-06), XP002199142 comp.compression Retrieved from the Internet: <URL:http://groups.google.com/groups?selm= 4u8ee0%24oef%40netline-fddi.jpl.nasa.gov&o utput=gplain> 'retrieved on 2002-05-16! page 1, line 21 - line 32 ---	9,10
A	FR 2 158 693 A (DOCUMENTOR SCIENCES CORP) 15 June 1973 (1973-06-15) page 1, line 1 - line 4 page 2, line 9 - line 35 ---	1,11

-/--

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 01/28146

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	QUETZALCOATL BRADLEY, R. NIGEL HORSPOOL, JAN VITEK: "JAZZ: An Efficient Compressed Format for Java Archive Files" PROCEEDINGS OF THE CASCON'98 CONFERENCE, 'Online! 1998, XP002199143 Toronto, Canada Retrieved from the Internet: <URL:http://www.csr.uvic.ca/{nigelh/Publications/jazz.pdf> 'retrieved on 2002-05-16! the whole document -----	1-8,11

Form PCT/ISA/210 (continuation of second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 01/28146

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9957885 A	11-11-1999	US 6243480 B1	05-06-2001
		WO 9957885 A1	11-11-1999
		EP 1075757 A1	14-02-2001
		WO 0068875 A1	16-11-2000
US 5837986 A	17-11-1998	US 5399846 A	21-03-1995
		US 5304786 A	19-04-1994
		US 5635697 A	03-06-1997
		US 5504322 A	02-04-1996
		US 5796090 A	18-08-1998
		AU 637409 B2	27-05-1993
		AU 6392390 A	09-10-1990
		CA 2026387 A1	06-07-1991
		DE 69027762 D1	14-08-1996
		DE 69027762 T2	09-01-1997
		DE 69033063 D1	20-05-1999
		DE 69033063 T2	05-08-1999
		EP 0439682 A2	07-08-1991
		EP 0565738 A1	20-10-1993
		EP 0733991 A2	25-09-1996
		JP 2846442 B2	13-01-1999
		JP 3204793 A	06-09-1991
		US 5506697 A	09-04-1996
		US 5471533 A	28-11-1995
		US 5489158 A	06-02-1996
		US 5644408 A	01-07-1997
		US 5710419 A	20-01-1998
		US 5760382 A	02-06-1998
		US 5974202 A	26-10-1999
		US 5243655 A	07-09-1993
		US 6002491 A	14-12-1999
		US 5337361 A	09-08-1994
		US 5880453 A	09-03-1999
FR 2158693 A	15-06-1973	FR 2158693 A5	15-06-1973